Speichern in SFML

Um in SFML das Speichern umzusetzen, müssen wir als erstes Wissen, wie man Sachen auf der Festplatte speichert. Dafür nutzen wir die von C# vorgegebene System.IO Klasse. Also werden wir in MainScreen als erstes eines using Anweisung schreiben, damit wir diese verwenden können:

using System;

using System.Collections.Generic;

using System.Text;

using SFML.Window;

using SFML.Audio;

using SFML.Graphics;

using SFML.System;

using SFML\_GAME.entities;

using SFML\_GAME.framework;

using System.IO;

namespace SFML\_GAME.screens

{

class MainScreen : Screen

{

Um jetzt mit der Datei zu arbeiten, benutzen wir sogennante Streams. Diese sind Schnittstellen, um mit externen Daten zu arbeiten. In System.IO gibt es 2 Klassen die wir brauchen: StreamReader und StreamWriter.

StreamReader kann aus Dateien lesen.

StreamWriter kann neue Dateien erstellen und in diese Schreiben.

Also erstellen wir als erstes ein StreamWriter Objekt.

Beim Erstellen des StreamWriter Objekts, müssen wir den Dateinamen + eventuellen Pfad angeben.(siehe "Externe Ressourcen").

StreamWriter writer = new StreamWriter("savefile.txt");

Da die Datei "savefile.txt" noch nicht existiert, wird sie dadurch erstellt.

Um jetzt etwas in die savefile zu schreiben, nutzen wir die write-Methode von StreamWriter.

writer.Write("Hallo Welt!");

Nachdem wir alles geschrieben haben, was wir wollten, müssen wir den StreamWriter immer schließen, weil die Datei sonst "offen" bleibt.

writer.Close();

Nachdem wir das Programm ausgeführt haben, sollten wir im Debug-Ordner unseres Projektes die geschriebene Datei finden.

**Achtung**: Mit der hier gezeigten Weise wird der vorherigen Inhalt der savefile.txt immer gelöscht. Will man den Inhalt retten, so muss man ihn vorher lesen und abspeichern.

Um jetzt den Inhalt lesen zu können, benutzen wir den StreamReader auf die gleiche Weise. Nur statt write benutzen wir die ReadToEnd Methode, die uns den gesamten Inhalt als String zurückgibt.

StreamReader reader = new StreamReader("savefile.txt");

String inhalt = reader.ReadToEnd();

reader.Close();

Jetzt können wir also Strings auf die Festplatte schreiben und sie wieder lesen. Was jetzt noch fehlt ist Daten in Strings umwandeln zu können und zurück. Nehmen wir als Beispiel die Position einer Entity.

float x = example.graphic.Position.X;

float y = example.graphic.Position.Y;

Wir speichern die Werte erstmal in passende Variablen.

String save = x.ToString() + "x" + y.ToString();

Mit der toString() Methode, können wir die Zahlen in ein String konvertieren und wir trennen X und Y wert mit einem "x".

Also wenn example an der Position 200|300 wäre, dann würde der String "200x300" sein.

Diese String schreiben wir dann mit unserem Streamwriter in die savefile.

float x = example.graphic.Position.X;

float y = example.graphic.Position.Y;

String save = x.ToString() + "x" + y.ToString();

StreamWriter writer = new StreamWriter("savefile.txt");

writer.WriteLine(save);

writer.Close();

Jetzt müssen wir beim Laden den String dementsprechend aufteilen und die Werte herauslesen. Die Split() Methode ist dafür gut geeignet.

StreamReader reader = new StreamReader("savefile.txt");

String inhalt = reader.ReadToEnd();

reader.Close();

String PosX = inhalt.Split('x')[0];

String PosY = inhalt.Split('x')[1];

Split() gibt dir von einem String einen Array zurück, wobei das Zeichen in den Klammern ein Trennzeichen darstellt, welches den String aufteilt.

Aus "300x200" wird also {"300","200"} in einem Array. Das [0] und [1] hinter den Ausdrücken greift jeweils das erste und zweite Element des Arrays ab, also die "300" und die "200".

Jetzt müssen wir aus den Strings wieder float Werte machen, dazu nutzen wir System.Convert.ToSingle().

float x = System.Convert.ToSingle(PosX);

float y = System.Convert.ToSingle(PosY);

example.graphic.Position = new Vector2f(x, y);

**Achtung**: Hier wäre eine Überprüfung Sinnvoll, ob der Inhalt der Savefile wirklich so aussieht wie wir denken und nicht anders. Sonst könnte es hier zu abstürzen kommen.